OPTIMIZATION METHOD FOR QUANTUM COMPUTING PROCESS

Alexandre Blais

COPYRIGHT NOTICE

REFERENCE TO APPENDIX ON COMPACT DISC

[0002] An Appendix containing a computer program listing is submitted on a compact disc, which is incorporated by reference herein in its entirety. The total number of compact discs including duplicates is two. Each of the compact discs includes a file named "anneal.txt", which was created February 7, 2001 and has a size of 19,285 bytes and file named "swap.txt", which was created February 6, 2001 and has a size of 11,791 bytes.

BACKGROUND

Field of the Invention

[0003] This invention relates to quantum computing and particularly to methods for reducing the required coherence time in a quantum computer performing a computation such as a quantum Fourier transform or any quantum network.

Description of Related Art

[0004] Research on what is now called quantum computing traces back to Richard Feynman, (R. Feynman, Int. J. Theor. Phys., 21, 467-488 (1982)). Feynman noted that quantum systems are inherently difficult to simulate with conventional computers but that observing the evolution of a quantum system could provide a much faster way to

solve some computational problems.

[0005] Quantum computing generally involves initializing the quantum states of a set of qubits (quantum bits), allowing the quantum states of the qubits to evolve under the influence of quantum gates, and reading the qubits after they have evolved. A qubit is conventionally a system having two quantum states (typically two degenerate states), and the state of the qubit typically has non-zero probabilities of being found in either state. N qubits provide a system with state that is a combination of 2^N states. The quantum gates control the evolution of the distinguishable quantum states and define calculations corresponding to the evolution of the quantum state of the system. This evolution, in effect, performs 2^N simultaneous calculations. Reading the qubits after evolution determines the states of the qubits and the results of the calculations.

[0006] The presentation by Shor (P.W. Shor, "Polynomial-Time Algorithms for Prime Factorisation and Discrete Logarithms on a Quantum Computer," SIAM J. Comp., 26:1484, 1997) of an efficient quantum process for factorization has kindled great interest in developing of software for quantum computing. In particular, a lot of attention has focused on developing methods for efficient computations and error correction in quantum computers.

[0007] In parallel, great effort as been invested in developing physical implementations of quantum computers that meet the very stringent requirements needed for the coherent manipulation of quantum information. The first proposals for such quantum computers were based on trapped ions, cavity quantum electrodynamics systems, and NMR spectroscopy. NMR-based implementations of quantum computers have been successful at least for a limited number of qubits. See E. Knill et al., "An Algorithmic Benchmark for Quantum Processing," Nature, 404:368 (2000), which is hereby incorporated by reference in its entirety. However, the inherent limitations of NMR-based quantum computers have motivated the search for more scalable designs.

[0008] The high level of expertise available in solid-state based technologies establishes solid-state systems as a leading candidate for the realization of a useful (several thousands of qubits) quantum computer. Solid-state quantum computers have

been proposed including Josephson junctions, quantum dots, or spin resonance transistors as qubits. Recent experimental success at coherently manipulating a solid-state qubit, for example, as described by Y. Nakamura et al., "Coherent Control of Macroscopic Quantum States in a Single-Cooper Pair Box," Nature (London), 398:786, 1999, gives good confidences that solid-state quantum computers are practical. However, the large numbers of degrees of freedom in solid-state quantum computers typically cause such quantum computers to suffer from short coherence times. (The coherence time is the time during which a quantum state of the quantum computer can evolve before external influences interfere with the quantum states of the qubits.) To take full advantage of the computational power of a solid-state quantum computer, optimization of the software for the specific quantum hardware will be critical. In particular, optimized software that reduces the total coherence time required for a computation will determine whether the quantum computer can complete a specific calculation.

SUMMARY

[0009] In accordance with the invention, fundamental gates necessary for a quantum Fourier transform and other quantum networks are built from physically realizable operations, each of which takes a specific time to complete. The fundamental gates and swap operations have commutation relationships that permit optimizations in the application of the gates and quantum computation processes employing the gates. In the case of the quantum Fourier transform algorithm, an order $O(N^2)$ speedup is obtained over the conventional ordering of the fundamental gates. Such improvements are considerable in the context of the physical realization of quantum algorithms where coherent manipulations are limited to a short coherence time.

[0010] One specific embodiment of this invention is a method for reducing required coherence time for a quantum computation. This is accomplished by constructing a second series of operations from a first series by changing the execution order of the commuting operations in a way maximizing the number of operations that can be performed simultaneously. This reduces the time required for a quantum computing

device to complete the second series of operations. Changing the execution order of the commuting operations can enable simultaneous execution of operations in the second series and/or can eliminate the need for some swap operations.

[0011] In accordance with another aspect of the invention, a method for performing a swap operation in a quantum computing device reduces the required computation time by simultaneously performing fundamental operations that commute.

BRIEF DESCRIPTION OF THE DRAWINGS

[0012] Fig. 1 is a block diagram of a multi-qubit quantum register suitable for computations described further herein.

[0013] Fig. 2 illustrates a network realizing a quantum Fourier transform on a 4-qubit register.

[0014] Figs. 3A and 3B respectively illustrate use of sequential and simultaneous swap operations to provide controlled interactions between non-adjacent qubits.

[0015] Fig. 4 illustrates a network performing a quantum Fourier transform using a 4-qubit register and simultaneous operations including swap operations.

[0016] Fig. 5 illustrates a quantum Fourier transform network for an array of four qubits allowing for nearest-neighbor interactions and simultaneous operations.

[0017] Fig. 6 illustrates a process for constructing an improved quantum Fourier transform network for any number of qubits.

[0018] Fig. 7 illustrates a further optimization of a network for a quantum Fourier transform.

[0019] Fig. 8 is a graph comparing the time costs of conventional quantum computing processes and optimized quantum computing process in accordance with the invention.

[0020] Use of the same reference symbols in different figures indicates similar or identical items.

DETAILED DESCRIPTION

[0021] In accordance with an aspect of the invention, a one-dimensional array of qubits that allows for nearest-neighbor interactions performs simultaneous operations on distinct qubits and thereby shortens the time required for quantum networks such as quantum Fourier transforms. For illustrative purposes, a particular solid-state quantum register suitable for implementation of quantum computational processes in accordance with the invention is described below. However, such quantum computational processes as described herein can be implemented in other quantum computing structures including other solid-state quantum registers that exist or may be developed and other types of quantum computers such as the NMR-based systems.

[0022] Fig. 1 illustrates an example of a solid-state multi-qubit register 100 containing a linear array of qubits. Quantum registers such as register 100 are further described in co-owned U.S. patent app. Nos. 09/452,749 and 09/479,336, which are hereby incorporated by reference in their entirety. Register 100 includes qubits based on Josephson junctions 130-0 to 130-(N-1) that are at the interfaces between a superconducting bank 110 and respective mesoscopic, superconducting islands 120-0 to 120-(N-1). A d-wave superconductor, for example, a high-Tc superconductor such as $YBa_2Cu_3O_{7-x}$ or any superconductor, in which the Cooper pairs are in a state with non-zero orbital angular momentum makes up one or both of bank 110 and islands 120-0 to 120-(N-1).

[0023] A Josephson junction having a d-wave superconductor on one or both sides has ground state current that is doubly degenerate. In particular, the ground state current at each of the Josephson junctions 130-0 to 130-(N-1) is non-zero and either clockwise or counter-clockwise. The ground state current at the ith Josephson junction has two basis quantum states |CWi> and |CCWi> respectively corresponding to clockwise and counterclockwise currents. Generally, each qubit is in a ground state that is a linear combination of the current states. For example, a state of the first qubit is a combination a*|CW0> + b*|CCW0>, where a and b are complex numbers. The two basis states |CWi> and |CCWi>, for each value i from 0 to (N-1), can be arbitrarily

assigned respective binary values 0 and 1.

[0024] The quantum state of register 100, which is the combination of N qubit states, is represented as |N-1, ..., 0>. State |N-1, ..., 0> has 2^N different ground state current configurations according to whether the current at each qubit is clockwise or counterclockwise, but generally state |N-1, ..., 0> is a combination of the different current states and does not have a definite current configuration. The N-qubit states having definite current configurations (and therefore corresponding to definite binary values) are designated herein as |x> or |y>, where x and y are N-bit binary values between 0 and 2^N-1. Accordingly, Equation 1 gives a general representation of the state |N-1, ..., 0>.

Equation 1: $$|N-1,...,0\rangle = \sum_{y=0}^{2^N-1} a_y |y\rangle$$

In Equation 1, coefficients a_y are generally complex numbers.

[0025] For calculations, register 100 is cooled to a low temperature (e.g., less than about 10 °K) to make bank 110 and islands 120-0 to 120-(N-1) superconductive and to suppress thermal sources of decoherence that would disturb the states of the qubits. Initial states of the qubits are then established, for example, by passing a current through bank 110. The initial state depends on the direction of current in bank 110, the crystal orientations of bank 110 and islands 120-0 to 120-(N-1), the nature and shape of the interface between bank 110 and islands 120-0 to 120-(N-1), and any externally applied magnetic fields.

[0026] Another way to initialize a qubit is to measure the state, i.e., determine whether the current at the junction is clockwise or counterclockwise. The measurement forces the qubit into a definite state corresponding to the measured value. If the measurement provides the desired result, the computation can start. If the desired state was not obtained, a NOT gate applied to the qubit can invert the qubit's state, and then a subsequent measurement acts as a kind of error correction to ensure that the NOT gate

was realized correctly. To perform a useful computation, the initial state must be known and typically has a definite binary value. Usually, the initial state is chosen to have the binary value 0, that is, all qubits are in state corresponding to 0.

[0027] The quantum calculation is performed as the initial state evolves under the influence of various interactions on the states. These interactions are commonly referred to as quantum gates. In quantum register 100, single electron transistors (SETs) 150-2 to 150-N are between neighboring islands 120-1 to 120-N. When SETs 150-2 to 150-N are off, the states |0> to |N-1> of the qubits evolve separately. When one or more SETs 150-2 to 150-N are on, SETs 150-2 to 150-N create controllable entanglements of the quantum states of neighboring qubits. Register 100 is one-dimensional in that entanglements can only be created between neighboring qubits along a single line.

[0028] When the quantum calculation is complete, the current states of the qubits are observed or read to determine results. For reading the results, SETs 140-0 to 140-(N-1), which are between ground and respective islands 120-0 to 120-(N-1), can be turned on to "freeze" the respective current states of the qubits. The current states of the qubits can then be determined with a measuring device (not shown). A magnetic force microscope (MFM) tip or a superconducting quantum interferometer device (SQUID) loop, for example, can measure the magnetic moments at the individual Josephson junctions to determine the current states.

[0029] A limitation of a quantum register is the coherence time of the qubits. The coherence time generally indicates the time during which the quantum state of a qubit can evolve before external factors disrupt the quantum state. A quantum calculation must be completed with the coherence time. Accordingly, one goal of quantum computing is to minimize the time costs of the process or network that performs a quantum calculation.

[0030] One useful quantum calculation is a quantum Fourier transform. The quantum Fourier transform is a quantum generalization of the classical Fourier transform. In the field of quantum computing, quantum Fourier transforms are the basis

for most know quantum computations. In particular, quantum Fourier transforms are used in Shor's factorization algorithm (and were developed by Shor for this very purpose). The quantum Fourier transform acts on the state |N-1, ...,0> of an N-qubit register as shown in Equation 2.

Equation 2: $$QFT_N : |x\rangle \rightarrow \frac{1}{2^{N/2}} \sum_{y=0}^{2^N-1} e^{2\pi i \frac{xy}{2^n}} |y\rangle$$

In Equation 2, state |x> and each state |y> has definite current configuration (i.e., correspond to definite N-bit values x and y), and the summation over states |y> is a summation over all definite current states (i.e., all N-bit values y). The quantum Fourier transformation of Equation 2 can be performed on a N-qubit register 100 using two basic quantum gates: a one-qubit gate A_i acting on the state initially corresponding to the ith qubit and a two-qubit gate B_{jk} acting on the states initially jth and kth qubits.

[0031] The one-qubit gate A_i (also known as the Hadamard transformation) is shown in Equation 3.

Equation 3: $$A_i = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

In Equation 3, the basis vectors for the operator are the two basis states |CWi> or (0 1) and |CCWi> or (1 0) of the ith qubit. The one-qubit gate can be implemented in quantum register 100 as described below.

[0032] Equation 4 corresponds to the two-qubit gate B_{jk}.

Equation 4: $$B_{jk} = \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & e^{i\theta_{jk}} \end{pmatrix}$$

In Equation 4, the basis vectors for the operator are the four basis states created from the cross product of the basis states |CWj> and |CCWj> of the jth qubit with the basis

states |CWk> and |CCWk> of the kth qubit. Coefficients j and k specify the pair of qubits on which to apply the quantum gate B_{jk}. The angle θjk is defined as $\pi/2^{k-j}$ and so is determined by the "distance" between the qubits on which quantum gate B_{jk} is applied. The two-qubit gate Bjk can be implemented in quantum register 100 as described further below.

[0033] Equation 5 indicates a sequence of quantum gates that when implemented in a 4-qubit register, performs the quantum Fourier transform of Equation 1 on the state of the quantum register.

Equation 5: $QFT_4 = A_3\, B_{23}\, A_2\, B_{13}\, B_{12}\, A_1\, B_{03}\, B_{02}\, B_{01}\, A_0$

[0034] Fig. 2 illustrates the quantum network 200 corresponding to Equation 5 and specifically illustrates the quantum gates acting on particular qubits. This quantum Fourier transform on a four-qubit register thus uses ten quantum gates, but more generally, computing an N-qubit quantum Fourier transform QFT_N uses N one-qubit operations and N(N-1)/2 two-qubit operations. The ten quantum gates act on respective qubits during ten time intervals 201 to 210. As described further below, the intervals 201 to 210 are generally not equal in duration. The duration of each of intervals 201 to 210 depends on the associated quantum gate and the underlying quantum register implementing the associated gate.

[0035] Optimizing a given computational process generally requires expressing the computational process in terms of the elementary set of gates of the quantum architecture in use. For example, Equations 6, 7, and 8 provide a universal set of operators X(θ), Z(ϕ), and CP(ζ).

Equation 6: $$X(\theta) = e^{\frac{-i\sigma_x\theta}{2}}$$

Equation 7: $$Z(\phi) = e^{\frac{-i\sigma_z\phi}{2}}$$

Equation 8: $$CP(\varsigma) = e^{\frac{-i\sigma_Z \otimes \sigma_Z \varsigma}{2}}$$

In Equations 6, 7, and 8, σ_X and σ_Z are Pauli matrices. Operators $X(\theta)$ and $Z(\phi)$ are single qubit operators and can be indexed according to the qubit operand. Operator $CP(\zeta)$ is a two qubit operator and require a pair of indices to identify the qubit operands.

[0036] Operators $X(\theta)$, $Z(\phi)$, and $CP(\zeta)$ correspond to the elementary set of gates of many solid-state designs and can be implemented in NMR-based quantum computers. See N.A. Gershenfeld and I.L. Chuang, "Bulk Spin-Resonance Quantum Computation," Science, 275:351, 1997. Implementation of Operators $X(\theta)$, $Z(\phi)$, and $CP(\zeta)$ in a quantum register such as illustrated in Fig. 1 is described in the paper of A. Zagoskin and A. Blais, "Operation of universal gates in a solid-state quantum computer based on clean Josephson junctions between d-wave superconductors' Phys. Rev. A 61 :042308 (2000)" which is hereby incorporated by reference in its entirety.

[0037] The elementary operations $X(\theta)$, $Z(\phi)$, and $CP(\zeta)$ implement the gate Bjk (on two adjacent qubits j and k) and the gate s Aj (on qubit j) as indicated in Equations 9 and 10.

Equation 9: $$A_j = i\, Z_j(\pi/2)\, X_j(\pi/2)\, Z_j(\pi/2)$$

Equation 10: $$B_{jk} = e^{i\theta_{k-j+1}} Z_j(\theta_{k-j+1}) Z_k(\theta_{k-j+1}) CP_{jk}(\theta_{k-j+1})$$

In Equation 10, the phase $\exp\{i\theta_{k-j+1}\}$ depends on which qubits j and k are the operands of operator Bjk but is independent of the states of qubits j and k. The phase $\exp\{i\theta_{k-j+1}\}$ is thus an unimportant global phase factor and can be ignored.

[0038] The network of Fig. 2 requires applying gate Bjk to nonadjacent qubits. However, a one-dimensional array of qubits generally limits interaction between qubits to nearest-neighbor couplings, e.g., in quantum register 100 of Fig. 1, each of SETs 130-1 to 130-(N-1) creates controlled a entanglement between two adjacent qubits.

Accordingly, applying gate Bjk to nonadjacent qubits entails swapping recursively the states of adjacent qubits so that the states move to adjacent qubits for interaction.

[0039] Quantum bits initially at locations j and k require |j-k|-1 swap operations to bring the qubits together and another |j-k|-1 swap operations to return the quantum bits to their original locations. For example, Fig. 3A shows a network implementing B_{jk} when j is 0 and k is 3. Two swaps during time intervals 301 and 302 bring the state from qubit 3 to qubit 1, which is adjacent to qubit 0. The quantum gate B is during time 303 while the states are in adjacent qubits, then two swaps during times 304 and 305 return the evolved state from qubit 1 to qubit 3.

[0040] A swap SW_{rs} between qubits at positions r and s respectively is usually realized by a sequence of controlled-NOT (CN) gates as shown in Equation 11.

Equation 11: $$SW_{rs} = CN_{rs}\, CN_{sr}\, CN_{rs}$$

In accordance with an aspect of the invention, the CN gate can be realized by a sequence of 7 elementary gates as shown in Equation 12.

Equation 12:

$$CN_{rs} = e^{-i3\pi/4}\, X_s(3\pi/2)\, CP_{rs}(\pi/2)\, Z_s(\pi/2)\, X_s(\pi/2)\, Z_s(\pi/2)\, Z_r(\pi/2)\, CP_{rs}(\pi/2)$$

As a result, a single swap operation SW_{rs} requires 21 elementary gates (time steps). This sequence in Equation 12 is shorter than that normally used for a CN gate. The main difference between Equation 12 and prior proposals is the use of two 2-qubit gates (CP) instead of one to realize the CN gate.

[0041] The number of elementary operations required for a swap can be significantly reduced using the commutation relations between the elementary gates and the symmetry of the Controlled-NOT gate. More particularly, removing redundant gates can reduce the number of time steps for the sequence of Equation 11. Equation 13 indicates a swap sequence implemented in 15 elementary operations.

Equation 13:

$$SW_{rs} = e^{-i\pi/4} X_s(3\pi/2)\, CP_{rs}(\pi/2)\, Z_s(\pi/2)\, X_s(\pi/2)\, [Z_s(\pi/2)\, X_r(\pi/2)]\, CP_{rs}(\pi/2)$$
$$Z_r(\pi/2)\, [X_r(\pi/2)\, X_s(\pi/2)]\, CP_{rs}(\pi/2)\, Z_s(\pi/2)\, X_s(\pi/2)\, [Z_s(\pi/2)\, Z_r(\pi/2)]$$

Furthermore, performing operations on different qubits simultaneously further reduces the number of time steps used for moving qubits through the register. Indeed, since operations on distinct qubits commute, the gates in square brackets in Equation 13 can be performed simultaneously, reducing the number of time steps required to complete a swap to twelve.

[0042] Further, instead of only swapping qubit j toward qubit k, as in Fig. 3A, simultaneously swapping qubits j and k with their neighbors as in Fig. 3B further reduces the number of time steps. Accordingly, the number of time steps used to juxtapose qubit states initially in the jth and kth qubits and return the interacted states to their original position is reduced to $2\lceil |j-k|/2 \rceil$, where $\lceil x \rceil$ is the smallest integer larger than x.

[0043] An additional simplification of the computational process is that states of qubits that have been juxtaposed don't have to be moved back to their original location. Instead, once two qubit states have been brought together and interacted, the next reorganization should be done in a way optimizing the realization of the following quantum gates. The location of the qubit states in the quantum register can be tracked classically, and bits can be reorder as required either during the initialization of qubits or when interpreting results read from the quantum register.

[0044] The above simplifications of the quantum computation reduces the number of physical time steps from $42(|j-k|-1)$ to $12\lceil |j-k|/2 \rceil$ for a single swap sequence to juxtapose the jth and kth qubits.

[0045] Fig. 4 shows an optimized quantum Fourier transform QFT_4 400 that reduces the number of swap operations in the manner described above. Additionally, the necessary swap operations are optimized using simultaneous swap operations and an efficient reordering of the qubits. In particular, initial states S0, S1, S2, and S3 of

respective qubits Q0, Q1, Q2, and Q3 respectively correspond to the resulting states S0', S1', S2', and S3', which are in qubits Q1, Q2, Q0, and Q3 respectively. The process of Fig. 4 includes ten logical gates and four swaps, which require 54 time steps. In evaluating the time cost of a given quantum network, when simultaneous operations are performed, the number of steps of the most time consuming operation is used. Without optimization, this transformation would require eight swaps in addition to the ten logical gates for a total of 126 physical time steps.

[0046] The computational process of Fig. 4 can be further optimized, in terms of the number of time steps, using reordering and simultaneous execution of gates that commute. In particular, 1-qubit gates such as gate Ai commute with the swap operation. The gate Ai is performed on the qubit containing the evolved state originally corresponding to the ith qubit and accordingly may be performed on a different qubit after a swap. A 1-qubit gate can be performed simultaneously with a swap only if both are not acting on the same qubit. A 2-qubit gate B_{jk} commutes with a swap only if they are not acting on the same qubit(s). For example, in Fig. 4, operation A_2 is during time interval 403 when the evolved state S2 is the state of the qubit Q2. Accordingly, operation A_2 is performed on qubit Q2. A swap operation during time interval 404 swaps state S1 from qubit Q1 into qubit Q2 and swaps state S2 from qubit Q2 into qubit Q1. As a result, states S3 and S1 will be adjacent in the register for operation B_{13}. By changing the order of application of the swap (interval 404) and A_2 (interval 403), A_2 and B_{13} can now be performed simultaneously. This is shown in the intervals 503 and 504 of figure 5.

[0047] Communitivity of other operators permit similar time savings. Specifically, as indicated by the commutators in Equations 14,15, and 16, gate A_i commutes with gate A_j if i is not equal to j, gate A_i commutes with gate B_{jk} if i is not equal to j or k, and gate B_{jk} commutes with B_{rs} for all j, k, r, and s.

Equation 14:	$[Ai,Aj] = 0$	if $i \neq j$
Equation 15:	$[Ai,Bjk] = 0$	if $i \neq j$ or k
Equation 16:	$[Bjk,Brs] = 0$	for all j, k, r, and s.

These commutation relations allow permutations of the order of the logical operations of the computational process of Fig. 4. For example, in Fig. 4, swap operations during interval 408 put the states evolved from states S0 and S3 in adjacent qubits Q1 and Q2 for operation B_{03} during interval 409. A subsequent swap during time interval 410 moves the state evolved from S1 into qubit Q2 for operation B_{01} is during interval 411 after operations B_{02} and B_{03}. In this ordering of operations, the swap operation of interval 410 undoes one of the swap operations of interval 408.

[0048] The process of Fig. 5 places operation B_{01} before operations B_{02} and B_{03} during an interval 507. This reordering eliminates a swap operation because the states evolved from states S0 and S1 are in adjacent qubits Q1 and Q2 for operation B_{01}. Of the swap operations of interval 408 in Fig. 4, the process of Fig. 5 eliminates one and performs the other during interval 506 simultaneously with operation A_1. Accordingly, the reordering further shortens the time required for the process of Fig. 5.

[0049] Fig. 6 illustrates a process for constructing an improved network for quantum Fourier transforms (QFT) on n qubits. In Fig. 6, the operations in block 630 implement an efficient QFT on three qubits. Adding four logical gates A_3, B_{23}, B_{13}, and B_{03} and three swaps to the QFT on three qubits provides a QFT on four qubits, which is contained in block 640. The added gates and swaps that provide the QFT_4 of block 640 add 29 time steps to the QFT_3 of block 630. Adding 5 logical gates A_4, B_{34}, B_{24}, B_{14}, and B_{04} and four swaps provides a QFT on five qubits. These added gates and swaps add a further 29 time steps.

[0050] Using this construction of Fig. 6 recursively on the operation network for QFT_{n-1} provides an optimized network for QFT_n. Constructing QFT_n from QFT_{n-1} requires addition of n logical gates A_{n-1}, $B_{(n-2),(n-1)}$, ... $B_{0,(n-1)}$ and n-1 swaps between qubits n-1 and n-2, qubits n-2 and n-3, ... and qubits 1 and 0, the swaps being interleaved with the added two qubit operations B. However, the number of added time steps is still 29, and the number of time steps required for this network construction of QFT_n is 8+29(n-2) ($\approx O(n)$)for n>2. The number of time steps required for a straitforward conventional implementation of a network for QFTn is $10n-11n^2+4n^3$ ($\approx O(n^3)$). Accordingly, the networks disclosed here provide significant performance

gains.

[0051] Fig. 8 shows the time cost of quantum Fourier transforms as a function of the number of qubits for up to 300 qubits on a logarithmic scale for both improved networks (black circles) and conventional non-improved networks (open squares). The improved networks were obtained numerically using the method disclosed above. In particular, grouping and parallel execution of operations that commute while not acting on similar qubits can maximize performance of parallel operations to minimize the required time for a given network. Such optimizations can be performed efficiently, taking a fews minutes for a 300 qubits network on a conventional desktop computer. The file "swap.txt" in the CD appendix contains a listing for the program that performs the automated optimization of QFTn.

[0052] For very small networks, both curves in Fig. 6 coincide while, for larger networks, it is clear that the numerical data correspond to circuits which are much more efficient. From the logarithmic plot of the numerical data, we obtain a slope of 1.08 ± 0.01 for networks involving more than 10 qubits. This confirms that the quantum Fourier transform can be implemented in O(n) time steps on n quantum bits. This corresponds to a speedup of order $O(n^2)$ compared to non-optimized networks. Efficient use of swaps and massive (classical) parallelism are responsible for this speedup. Indeed, speedup by a factor of O(n) can be seen to come from the fact that $O(n^3)$ swaps are necessary in non optimized circuits while only $O(n^2)$ in the optimized case. The other factor of O(n) comes from the fact that up to n simultaneous operations can be realize on n qubits. As in the case of classical parallel computers, this provides a speedup of order O(n).

[0053] The networks having the construction of Fig. 6 can be further shortened by permuting the logical operations to reduce the number of swaps and to maximize parallel performance of operations. Fig. 7 illustrates QFT_5 after further optimization.

[0054] Minimizing the time required for the network corresponds for solving a constrained optimization problem and has many similarities to the problem of placement occurring in VLSI related technologies. In placement, one seeks to arrange

the components of a classical circuit in a way minimizing the length of interconnecting wires and area of the circuit. Heuristitics like simulated annealing or tabu search are known to give good results for such problems. The problem of optimizing a network for a quantum calculation is very similar but with the additional complication that reordering two logical operations at a given location in a circuit will change the sequence and possibly the number of swaps needed at all further points in this circuit. The file "anneal.txt" in the CD appendix contains a listing for a program that performs simulated annealing to optimize a network for a QFTn. As this is a heuristic process, the program will not provide optimal solutions but solutions which are close to the optimal solution.

[0055] Although the invention has been described with reference to particular embodiments of quantum computers and a particular algorithm, the description is only an example of the invention's application and should not be taken as a limitation. Although much of the above disclosure was directed at examples implementing quantum Fourier transforms (QFTs), other quantum computing procedures can similarly benefit from the procedures described. Additionally, although quantum computations where disclosed for a system employing a linear arrangement of qubits, embodiments of the invention are also applicable to other quantum computing architectures including but not limited to two-dimensional arrays of qubits with nearest neighbor interactions. Various other adaptations and combinations of features of the embodiments disclosed are within the scope of the invention as defined by the following claims.